NETWORK CODED MULTIPATH FOR 5G

Muriel Médard

Cecil H. Green Professor of Electrical Engineering and Computer Science
Network Coding and Reliable Communications Group
Research Laboratory for Electronics
Massachusetts Institute of Technology.
Collaborators

- MIT: Jason Cloud, Flavio du Pin Calmon, Dina Katabi, David Karger, Weifei Zeng
- MIT Lincoln Lab: Linda Zeger (now Aurora)
- Duke University: Ahmad Beirami, Robert Calderbank
- Stanford: Sachin Katti
- CMU: Bernhard Haeupler
- Caltech: Michelle Effros, Tracey Ho
- UCLA: Giovanni Pau, Jun Shi
- University of California Berkeley: Alessandro Chiesa, Raluca Popa
- SUNY Buffalo: Michael Langberg
- Hamilton Institute, Maynooth University, Ireland: Mark Christiansen, Ken Duffy
- Technical University of Munich, Germany: Ralf Koetter
- Technical University of Dresden, Germany: Frank Fitzek
- Budapest University of Technology and Economics, Hungary: Hassan Charaf, Marton Sipos, Aron Szabados, Thomas Toth
- University of Porto, Portugal: Joao Barros, Luisa Lima, Paulo Oliveira, Tiago Vinhoza
- University of Aalborg, Denmark: Jonas Hansen, Jeppe Krisglund, Daniel Lucani, Morten Pedersen
- University of Auckland, New Zealand: 'Etuate Cocker, Ulrich Speidel
- National University of Singapore: Ben Leong, Fang Zhao
- University of Warsaw, Poland: Szymon Acedanski.
- Chinese University of Hong Kong: Sid Jaggi
- Ben Gurion University, Israel: Chen Avin, Asaf Cohen
- Steinwurf: Janus Heide, Peter Vingelmann
- Oracle: Minji Kim
- Namo Media: Tural Badirkhanlo
- AT&T: Supratim Deb
- DTS: Ton Kalker
- Air Force Research Laboratory: Keesook Han.
Directions

• Use coding to make a virtual network – coding is a means to create a synthetic, stable version
• Reduce control and state information, which cannot work in a variable, growing and heterogeneous setting
• We should become highly opportunistic and omnivorous
• Rapidly evolving domains
• Storage and networking are not separable
• Security has to be inherently considered.
Random Linear Network Coding (RLNC)

**Traditional Approach**

Data broken into pieces

All pieces needed to re-construct

Specific pieces

**Random Linear Network Coding**

Mixtures created *randomly* from pieces

Any mixtures will do
**Random Linear Network Coding (RLNC)**

Algebraic equations more efficiently input data into IP packets

An IP packet payload

```
001101001010001
```

Vector of elements of a finite field

4 packets

<table>
<thead>
<tr>
<th>Packet</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>ENCODE (using algebra)</td>
</tr>
<tr>
<td>$P_2$</td>
<td></td>
</tr>
<tr>
<td>$P_3$</td>
<td></td>
</tr>
<tr>
<td>$P_4$</td>
<td></td>
</tr>
</tbody>
</table>

Random linear combinations are highly likely to be recoverable

- $P_1 + 7P_2 + 4P_3 + 3P_4$
- $2P_1 + 8P_2 + 9P_3 + 5P_2$
- $8P_1 + 6P_2 + P_3 + 2P_4$
- $7P_1 + 2P_2 + 3P_3 + 4P_4$

RLNC Used in a Lossy Broadcast

Wireless network with 50% packet loss

Solve for missing data variables

Fewer data packet re-sends
Experimental Results in the Pacific using Coded TCP

Tunnel between Auckland and Rarotonga.

Composability: Redundancy When Needed

Over reliable network send minimum packets
(4 variables, 4 packets)

20% Loss Node

Re-encode → add a packet without decoding
(4 variables, 5 packets)

Needed packets arrive
(4 variables, 4 packets)

RLNC allows optimized packet sends at any node
Re-Encoding

Optimal and Dynamic Loss Compensation

Coding End-to-End Overhead = Cumulative Losses (37%)

37% UPFRONT Redundancy Added

Re-Coding Overhead = Single Worst-Case Loss (11%)

11% UPFRONT Redundancy Added
SDN with Recoding

Open vSwitch 5, an open-source virtual switch supporting the OpenFlow protocol

Multipath / Multicloud

Traditional RLNC Enables Stateless Communications

RLNC Enables Packet Networks
CTCP versus Multipath TCP Using Routing

MPTCP may potentially provide multi-path communication BUT:

- Difficult and complex scheduling at the source needed
- Round-robin scheduling is inefficient

CTCP provides multi-path communication throughput of 2.7 without complex scheduling at the source

Streaming through Multiple Interfaces

Server

Application layer

Divide & Schedule

Block 1

IP flow 1

p_1\ p_2\ p_3\ \ldots\ p_w

Block 2

IP flow 2

p_1\ p_2\ p_3\ \ldots\ p_w

Block 3

Media file: p_1\ p_2\ p_3\ \ldots\ p_w

Request block i

Client

Application layer

Combine

Block i

IP flow 1

Lte NIC

IP flow 2

WiFi NIC

Media player
Streaming through Multiple Interfaces

![Diagram of streaming through multiple interfaces]

Media file: $p_1 p_2 p_3 ... p_w p_1 p_2 p_3 ... p_w p_1 p_2 p_3 ... p_w ...$

**Server**

- Application layer
- Block $i$
- (Network) Encoder
- IP flow 1
- IP flow 2
- $p_1 + p_5$
- $p_2 + p_3$
- $p_1 + p_4$
- $p_4 + p_5$
- $p_3 + p_6$
- $p_2 + p_4$

**Client**

- Application layer
- Request block $i$
- (Network) Decoder
- IP flow 1
- IP flow 2
- $p_1 + p_4$
- $p_2 + p_3$
- $p_3 + p_6$
- $p_4 + p_5$

Media player
All received packets are ACK’ed

All packets within the MPTCP/NC coding window are coded together and pushed to different sub-flows

Each packet from TCP generates R linearly independent coded packets where each packet is generated from the set of packets within the TCP window

MPTCP/NC layer ACKs all new degrees of freedom received

Decoding only occurs at the MPTCP/NC layer

Each received packet is immediately forwarded to the TCP layer
Example of MPTCP with RLNC
Stabilize Below - SDN

(a) Network structure

(b) Multi-path network coding performance

Throughput [Mbps]

Loss Probability, Path 2 [%]

4x

4.5x

2.5x
5G and RLNC

Distributed Clouds
Distributed Clouds

Heterogenity (4 clouds)

- Clouds behave differently

Speed-Up (5 clouds)
What if Someone Pollutes the Cloud?

• We can detect the presence of attacks without a trusted authority
• Use codes borrowed from transmission to correct the erasures
• Distributed – there is no tracker and no way to bring it down

Trusted Storage over Untrusted Networks

Coded Mesh / Distributed Storage

Add or extract data on the fly

Adaptive, Mobile Network

S. Acedanski, S. Deb, Médard, M., and Koetter, R., “How Good is Random Linear Coding Based Distributed Networked Storage?” Workshop on Network Coding, Theory, and Applications (NETCOD) April 2005
B. Haeupler, Cohen, A., Avin, C., and Médard, M., “Network Coding Based Information Spreading in Dynamic Networks with Correlated Data”, accepted to IEEE Journal of Selected Areas in Communication
Dynamic Robustness and Repair

File made up of 16 chunks

Broken into 4 chunk pieces

Stored in 10 data center locations in the cloud

Three of the storage sites are randomly killed per round

How reliably can the data be reconstructed?

Example

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks,
  4 chunks each
Redundancy 33%
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

Decode
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
Reed-Solomon

File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

I/O Network: Intra-Rack Inter-Rack Processing

RS: 15 0* 15 Decode + Encode 15x15 matrix (new rack)

RLNC:

* May require some intra-rack transfer depending on structure
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

Mix (recode), Send 1 chunk
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

Mix (recode), Send 1 chunk
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

Mix (recode), Send 1 chunk
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

Mix (recode), Send 1 chunk
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

<table>
<thead>
<tr>
<th>I/O</th>
<th>Network: Intra-Rack</th>
<th>Inter-Rack</th>
<th>Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS:</td>
<td>15</td>
<td>0*</td>
<td>15</td>
</tr>
<tr>
<td>RLNC:</td>
<td>15</td>
<td>11</td>
<td>4</td>
</tr>
</tbody>
</table>

- Decode + Encode 15x15 matrix (new rack)
- Encode 4x4 matrices (4 times), and one 3x3 matrix

* May require some intra-rack transfer depending on structure
File made up of 15 chunks

Stored in 5 racks, 4 chunks each
Redundancy 33%

RLNC

<table>
<thead>
<tr>
<th>I/O</th>
<th>Network: Intra-Rack</th>
<th>Inter-Rack</th>
<th>Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS:</td>
<td>15</td>
<td>0*</td>
<td>15</td>
</tr>
<tr>
<td>RLNC:</td>
<td>15</td>
<td>11</td>
<td>4</td>
</tr>
</tbody>
</table>

* May require some intra-rack transfer depending on structure
Memory Consumption RS vs RLNC

- Reed-Solomon
  - Process 1 starts
  - Process 2 starts
  - Process 3 starts
  - Process 2 ends
  - Process 3 ends
  - Process 1 ends

- RLNC
  - Process 1 starts
  - 2nd packet (already recoded)
  - Process 1 ends
  - Process 1 recoding
Dynamic Robustness and Repair

No Coding

RS Coding

Network Coding
The Fog

- Edge wireless cloud, proximate peer-to-peer
- Use location as a type of security
- Use coding to ensure anonymity, resilience to pollution
- Use peers to watch over each other – watchdog approach
- Force peers to code correctly by colluding to keep nodes honest